LINUX JOURNAL™

Since 1994: The Original Magazine of the Linux Community

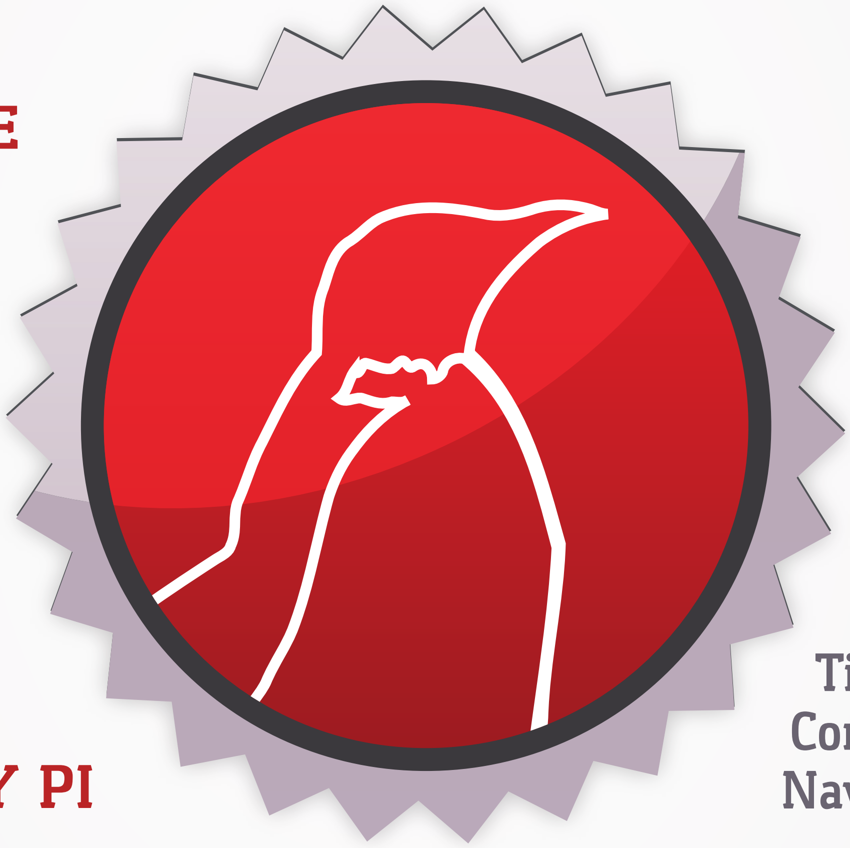
PHP FOR THE NON-DEVELOPER

INTRO TO MULTITENANT APPLICATIONS

DECEMBER 2014 | ISSUE 248 | www.linuxjournal.com

READERS' CHOICE AWARDS 2014

SEE HOW THE GNOME DESKTOP FARES IN A USABILITY STUDY



DISCOVER ROGUE WAPs WITH A RASPBERRY PI

+ Time-Saving Command-Line Navigation Tips

A LOOK AT **Power Shell History**

WATCH: ISSUE OVERVIEW

The Usability of GNOME

Jim Hall writes about his latest usability study of open-source software, doing a "deep dive" into the usability of the GNOME desktop.

JIM HALL

I work at a university, and one of our faculty members often repeats to me, "Software needs to be like a rock; it needs to be that easy to use." And, she's right. Because if software is too hard to use, no one will want to use it.

I recently spoke at GUADEC, the GNOME Users And Developers European Conference, and I opened my presentation with a reminder that GNOME is competing for mind share with other systems that are fairly easy for most people to use: Mac, iPad, Windows and Chromebook. So for GNOME to continue to be successful, it needs to be easy for everyone to use—experts and newcomers alike. And, that's where usability comes in.

So, what is usability? Usability is about the users. Users often are busy people who are trying to get things done. They use programs to be productive, so the users decide when a program is easy to use. Generally, a program has good usability if it is easy for new users to learn, easy for them to use, and easy for them to remember when they use the program again.

In a more practical view, average users with typical knowledge should be able to use the software to perform real tasks. The purpose of usability testing, therefore, is to uncover issues that prevent general users from employing the software successfully. As such, usability testing differs from quality assurance testing or unit testing, the purpose of which is to uncover errors in the program. Usability testing is not a functional evaluation of the program's features, but rather a practical determination of the program's operability.

Usability testing does not rely on

Usability cannot be addressed only at the end of a software development lifecycle. If you wait until the end, it is usually too late to make changes.

a single method. There are multiple approaches to implement usability practices, from interviews and focus groups to formal usability testing. Whatever method you use, the value of usability testing lies in performing the evaluation during development, not after the program enters functional testing when user interface changes become more difficult to implement. In open-source software development, the community of developers must apply usability testing iteratively throughout development. Developers do not require extensive usability experience to apply these usability practices in open-source software.

I prefer the formal usability test, and it's not hard. You can gain significant insight just by gathering a few testers and watching them use the software. With each iteration, usability testing identifies a number of issues to resolve and uncovers additional issues that, when addressed, will further improve the program's ease of use. Usability cannot be addressed only at the end of a software development lifecycle. If you wait until the end, it is usually too late to make changes.

How to Run a Usability Test—Usability Testing in GNOME

I recently worked with the GNOME Design Team to examine the usability of GNOME. This was an opportune moment to do usability testing in GNOME. The project was in the process of updating the GNOME design patterns: the gear menu, the application menu, selection mode and search, just to list a few. I assembled a usability test to understand how well users understand and navigate the new design patterns in GNOME. Here is how I did that.

The first step in planning a usability test is to understand the users. Who are they, and what tasks do they typically want to perform? With GNOME, that answer was easy. The GNOME Web site explains that "GNOME 3 is an easy and elegant way to use your computer. It is designed to put you in control and bring freedom to everybody." GNOME is for everyone, of all ages, for casual users and software developers.

From there, I worked with the GNOME Design Team to build a usability test for these users. The test also needed to exercise the GNOME design patterns. I compared the design patterns used by each GNOME program and decided five GNOME applications provided a reasonable representation of the design patterns:

1. gedit (text editor)

2. Web (Web browser)

3. Nautilus (file manager)

4. Software (similar to an app store)

5. Notes (a simple note-taking program)

Having decided on the programs, I wove a set of test scenarios that exercised the design patterns around tasks that real people would likely do. Designing the test scenarios is an important part of any usability test. You need to be very careful in the wording. It is too easy to "give away" accidentally how to do something just by using a particular turn of phrase. For example, one of my scenarios for Web asked testers to "Please make the text bigger" on a Web site—not to "Increase the font size", which would have hinted at the menu action they would need to use to accomplish the task.

Because I work on a university campus, I invited students, faculty and staff to participate in a usability study. Volunteers were selected without preference for gender, age group or level of experience. This reflects GNOME's preference to target a broad range of users. Each tester was given a $5 gift card to the campus coffee shop as a "thank you" for participating in the usability test.

In total, 12 testers participated in the usability test, representing a mix of genders and an age range spanning 18–74. Participants self-identified their level of computer expertise on a scale from 1 to 5, where 1 indicated "No knowledge" and 5 meant "Computer expert". No testers self-identified as either 1 or 5. Instead they filled a range between "2: I know some things, but not a lot", and "4: I am better than most", with an average rating of 3.25 and a mode of "3: I am pretty average."

Before each usability test session, every participant received a brief description of the usability study, explaining that this was a usability test of the software, not of them. This introduction also encouraged testers to communicate their thought process. If searching for a print action, for example, the participant should state "I am looking for a 'Print' button." Testers were provided a laptop running a "liveUSB" image of GNOME 3.12 containing a set of example files, including the text file used in the gedit scenario tasks. However, the USB image proved unstable for most programs in the usability test. To mitigate the stability issues, I rebooted the laptop into Fedora 20 and GNOME 3.10 to complete the scenario tasks for Web, Nautilus, Software and Notes. In testing GNOME, each participant used a separate guest account that had been pre-loaded with the same example files. These example files also included items unrelated to the usability test, similar to how real users keep a variety of documents, photos and other files on their computers, allowing participants to navigate these contents to locate files and folders required for the scenario tasks.

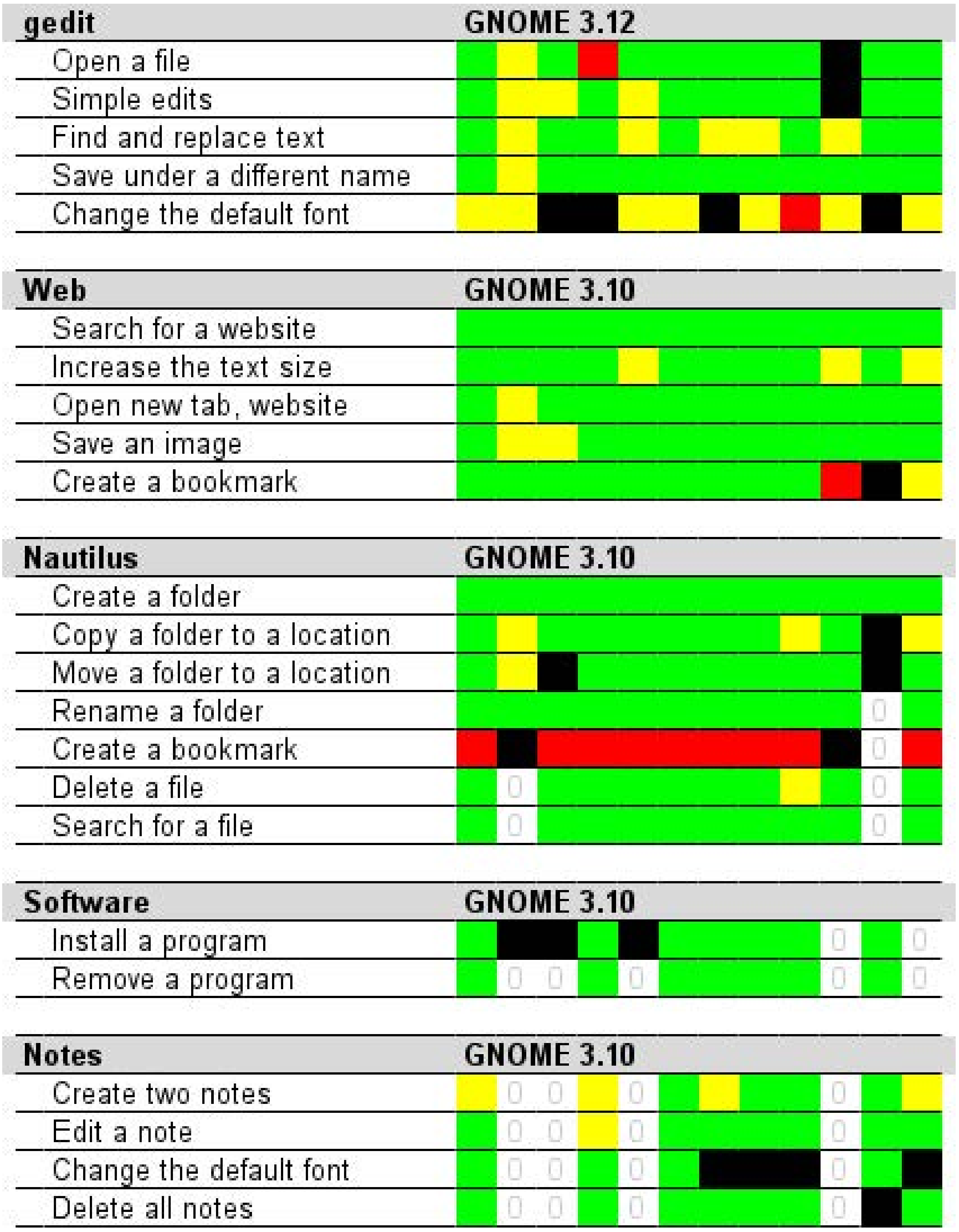
At the end of each usability test, I briefly interviewed the testers to probe their responses to the tasks and programs, to better understand what they were thinking during certain tasks that seemed particularly easy or difficult. Overall, the usability test included 23 scenario tasks, which most testers completed in 50–55 minutes.

My Usability Test Results

The value of a usability test is finding areas where the program is difficult to use, so developers can improve the interface in the next version. To do that, you need to identify the tasks that were difficult for most users to perform.

You easily can see the usability test results by using a heat map. This kind of visualization neatly summarizes the results of the usability test. In the heat map, each scenario task is represented in a separate row, and each block in the row represents a tester's experience with that task (Figure 1). Green blocks indicate tasks that testers completed with little or no difficulty, and yellow blocks signify tasks that presented moderate difficulty. Red boxes denote tasks where testers experienced extreme difficulty or where testers completed tasks incorrectly. Black blocks indicate tasks the tester was unable to complete, while white boxes indicate tasks omitted from the test, usually for lack of time.

The "hot spots" in the heat map show tasks that were difficult for testers.



gedit	**GNOME 3.12**											
Open a file	🟩	🟨	🟩	🟥	🟩	🟩	🟩	🟩	🟩	⬛	🟩	🟩
Simple edits	🟩	🟨	🟨	🟩	🟨	🟩	🟩	🟩	🟩	⬛	🟩	🟩
Find and replace text	🟩	🟨	🟩	🟩	🟨	🟩	🟨	🟨	🟩	🟨	🟩	🟩
Save under a different name	🟩	🟨	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Change the default font	🟨	🟨	⬛	⬛	🟨	🟨	⬛	🟨	🟥	🟨	⬛	🟨
Web	**GNOME 3.10**											
Search for a website	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Increase the text size	🟩	🟩	🟩	🟨	🟩	🟩	🟩	🟩	🟩	🟨	🟩	🟨
Open new tab, website	🟩	🟨	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Save an image	🟩	🟨	🟨	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Create a bookmark	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟥	⬛	🟨
Nautilus	**GNOME 3.10**											
Create a folder	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Copy a folder to a location	🟩	🟨	🟩	🟩	🟩	🟩	🟩	🟩	🟨	🟩	⬛	🟨
Move a folder to a location	🟩	🟨	⬛	🟩	🟩	🟩	🟩	🟩	🟩	🟩	⬛	🟩
Rename a folder	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	0	🟩
Create a bookmark	🟥	⬛	🟥	🟥	🟥	🟥	🟥	🟥	🟥	⬛	0	🟥
Delete a file	🟩	0	🟩	🟩	🟩	🟩	🟩	🟩	🟨	🟩	0	🟩
Search for a file	🟩	0	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	0	🟩
Software	**GNOME 3.10**											
Install a program	🟩	⬛	⬛	🟩	⬛	🟩	🟩	🟩	🟩	0	🟩	0
Remove a program	🟩	0	0	🟩	0	🟩	🟩	🟩	🟩	0	🟩	0
Notes	**GNOME 3.10**											
Create two notes	🟨	0	0	🟨	0	🟩	🟨	🟩	🟩	0	🟩	🟨
Edit a note	🟩	0	0	🟨	0	🟩	🟩	🟩	🟩	0	🟩	🟩
Change the default font	🟩	0	0	🟩	0	🟩	⬛	⬛	⬛	0	🟩	⬛
Delete all notes	🟩	0	0	🟩	0	🟩	🟩	🟩	🟩	0	⬛	🟩

Figure 1. Usability Heat Map

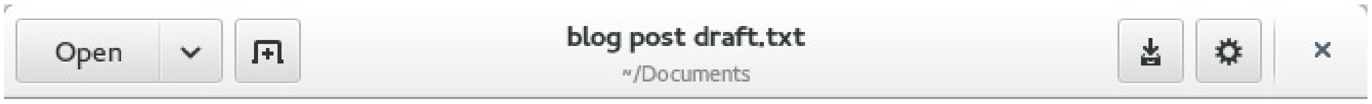


Figure 2. Header Bar for gedit, Showing the Gear Menu

1) Interestingly, all participants experienced significant issues with changing the default font in gedit (GNOME 3.12). A significant number of testers were unable to accomplish a similar task in Notes. In observing the tests, the testers typically looked for a "font" or "text" action under the gear menu. Many participants referred to the gear menu as the "options" or "settings" menu because of a previous affiliation with the gear icon and "settings" in other Mac OS X and Windows applications (Figure 2).

These participants expected that changing the font was an option in the program, and therefore searched for a "font" action under the gear or "options" menu. Part of this confusion stemmed from thinking of the text editor as though it were a word processor, such as Microsoft Word, which uses items in menus or on a toolbar to set the document font. This behavior often was exhibited by first highlighting all the text in the document before searching for a "font" action.

2) In the Nautilus file manager, testers also experienced serious difficulty in creating a bookmark to a folder. In GNOME, this task is usually achieved by clicking into the target folder then selecting "Bookmark this Location" from the gear menu, or by

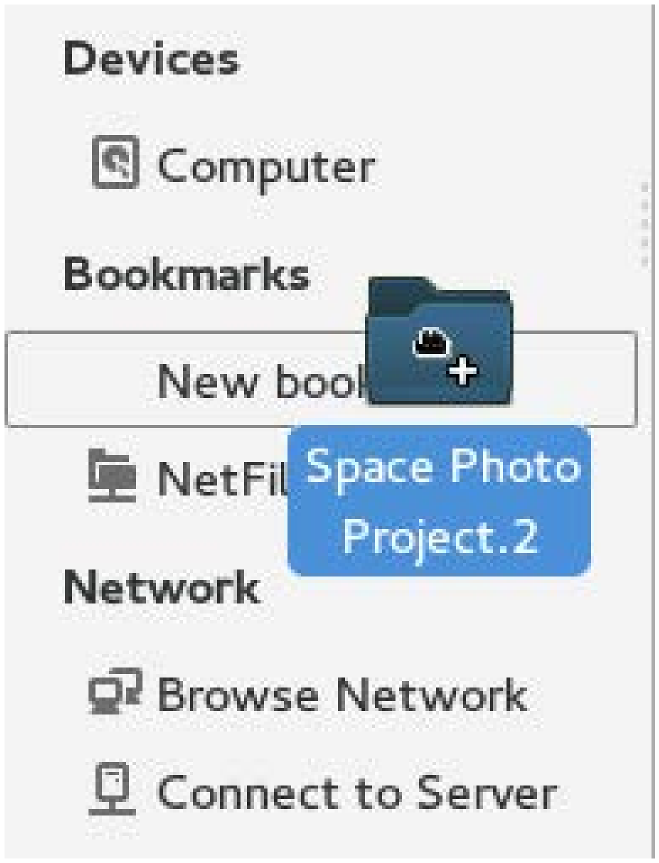


Figure 3. Dragging a Folder to Create a Bookmark in Nautilus

clicking and dragging the intended folder onto "Bookmarks" in the left pane (Figure 3).

However, testers initially addressed this task by attempting to drag the folder onto the GNOME desktop. When interviewed about this response, almost all participants indicated that they prefer to keep frequently accessed folders on the desktop for easier access. Most testers eventually moved the target folder into the Desktop folder in their Home directory, and believed they successfully completed the task even though the target folder did not appear on the desktop.

3) Testers also experienced difficulty when attempting "find and replace text" in gedit. In this task, testers employed the "Find" feature in gedit to search for text in the document. Experimenting with "Find", testers said they expected to replace at the same time they searched for text. After several failed attempts, testers usually were able to invoke the "Find and Replace" action successfully under the gear menu.

Although the overall GNOME desktop was not part of the usability test, many testers experienced difficulty with the GNOME "Activities" hot corner. In the GNOME desktop environment, the Activities menu reveals a view of currently running programs and a selection of available programs. Users can trigger the Activities menu either by clicking the "Activities" word button in the upper-left corner of the screen or by moving the mouse into that corner (the "hot corner"). Although testers generally recovered quickly from the unexpected "hot corner" action, this feature caused significant issues during the usability test.

General Issues

The open-source software usability challenge is cultural. To date, usability has been antithetical to open-source software philosophy, where most projects start by solving a problem that is interesting to the developer, and new features are incorporated based on need. Crafting new functionality takes priority, and open-source developers rarely consider how end users will access those features.

However, a key strength of open-source software development is the developer-user relationship. In open-source software projects, the user community plays a strong role in testing new releases. Unfortunately, testers cannot rely on the typical user-testing cycle to provide sufficient user interface feedback. Developers can learn a great deal simply by observing a few users interacting with the program and making note where testers have problems. This is the essence of a usability test.

Usability tests need not be performed in a formal laboratory environment, and developers do not need to be experts in order to apply usability testing methodology to their projects. Developers need only observe users interacting with the program for usability issues to become clear. A handful of usability testers operating against a prototype provides sufficient feedback to make informed usability improvements. And with good usability, everyone wins.■

Jim Hall is an advocate for free and open-source software, best known for his work on the FreeDOS Project. At work, Jim is the IT Director and IT Executive at the University of Minnesota Morris. In May, Jim received his Master's in Scientific and Technical Communication, studying the usability of open-source software.

Send comments or feedback via http://www.linuxjournal.com/contact or to ljeditor@linuxjournal.com.